

main ▾

...

## TinyTrackGPS / src / Display.cpp



RafaelReyesCarmona Final v0.5 ✓

History

1 contributor

282 lines (251 sloc) | 7.52 KB

...

```

1  /*
2  Display.cpp - A simple track GPS to SD card logger. Display module.
3  TinyTrackGPS v0.5
4
5  Copyright © 2019-2021 Francisco Rafael Reyes Carmona.
6  All rights reserved.
7
8  rafa.el.reyes.carmona@gmail.com
9
10   This file is part of TinyTrackGPS.
11
12   TinyTrackGPS is free software: you can redistribute it and/or modify
13   it under the terms of the GNU General Public License as published by
14   the Free Software Foundation, either version 3 of the License, or
15   (at your option) any later version.
16
17   TinyTrackGPS is distributed in the hope that it will be useful,
18   but WITHOUT ANY WARRANTY; without even the implied warranty of
19   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
20   GNU General Public License for more details.
21
22   You should have received a copy of the GNU General Public License
23   along with TinyTrackGPS. If not, see <https://www.gnu.org/licenses/>.
24  */
25
26  #include "Display.h"
27
28  Display::Display(Display_Type t):_screen(t){
29      if (_screen == SDD1306_128X64){

```

```

30     _width = 16;
31     _height = 8;
32     _offset = 0;
33 } else if (_screen == LCD_16X2 || _screen == LCD_16X2_I2C){
34     _width = 16;
35     _height = 2;
36     _offset = 0;
37 }
38 }
39
40 void Display::start(){
41     if (_screen == LCD_16X2){
42         #if defined(DISPLAY_TYPE_LCD_16X2)
43             // DEFINICION DE CARACTERES PERSONALIZADOS
44             byte alt[8] = {
45                 0b00000100,
46                 0b00001110,
47                 0b00011111,
48                 0b00000100,
49                 0b00000100,
50                 0b00000100,
51                 0b00000100,
52                 0b00000100,
53             };
54
55             byte ant[8] = {
56                 0b00001110,
57                 0b00010001,
58                 0b00010101,
59                 0b00010001,
60                 0b00000100,
61                 0b00000100,
62                 0b00001110,
63                 0b00000000,
64             };
65
66             byte sd[8] = {
67                 0b00001110,
68                 0b00010001,
69                 0b00011111,
70                 0b00000000,
71                 0b00000000,
72                 0b00010111,
73                 0b00010101,
74                 0b00011101,
75             };
76
77             byte hourglass_0[8] = {
78                 0b00011111,

```

```
79         0b00001110,
80         0b00001110,
81         0b00000100,
82         0b00000100,
83         0b00001010,
84         0b00001010,
85         0b00011111,
86     };
87
88     byte hourglass_1[8] = {
89         0b00011111,
90         0b00001010,
91         0b00001110,
92         0b00000100,
93         0b00000100,
94         0b00001010,
95         0b00001010,
96         0b00011111,
97     };
98
99     byte hourglass_2[8] = {
100         0b00011111,
101         0b00001010,
102         0b00001110,
103         0b00000100,
104         0b00000100,
105         0b00001010,
106         0b00001110,
107         0b00011111,
108     };
109
110     byte hourglass_3[8] = {
111         0b00011111,
112         0b00001010,
113         0b00001010,
114         0b00000100,
115         0b00000100,
116         0b00001010,
117         0b00001110,
118         0b00011111,
119     };
120
121     byte hourglass_4[8] = {
122         0b00011111,
123         0b00001010,
124         0b00001010,
125         0b00000100,
126         0b00000100,
127         0b00001110,
```

```

128         0b00001110,
129         0b00011111,
130     };
131     #endif
132     #if defined(DISPLAY_TYPE_LCD_16X2)
133     lcd = new LiquidCrystal(RS, ENABLE, D0, D1, D2, D3);
134     #endif
135
136     #if defined(DISPLAY_TYPE_LCD_16X2)
137     lcd->begin(_width, _height);
138
139     lcd->createChar(0, hourglass_0);
140     lcd->createChar(1, hourglass_1);
141     lcd->createChar(2, hourglass_2);
142     lcd->createChar(3, hourglass_3);
143     lcd->createChar(4, hourglass_4);
144     lcd->createChar(5, alt);
145     lcd->createChar(6, ant);
146     lcd->createChar(7, sd);
147     #endif
148 }
149
150 if (_screen == SDD1306_128X64) {
151     #if defined(DISPLAY_TYPE_SDD1306_128X64)
152     u8x8_SSD1306 = new U8X8_SSD1306_128X64_NONAME_HW_I2C(U8X8_PIN_NONE, SCL, SDA);
153     u8x8_SSD1306->begin();
154     u8x8_SSD1306->setFont(u8x8_font_7x14B_1x2_r); //u8g2_font_helvr10_tf
155     //u8x8_SSD1306->setFontRefHeightExtendedText();
156     //u8x8_SSD1306->setDrawColor(1);
157     //u8x8_SSD1306->setFontPosTop();
158     #endif
159 }
160 }
161
162 void Display::clr(){
163     if (_screen == LCD_16X2 || _screen == LCD_16X2_I2C) {
164         #if defined(DISPLAY_TYPE_LCD_16X2) || (DISPLAY_TYPE_LCD_16X2_I2C)
165         lcd->clear();
166         #endif
167     }
168     else if (_screen == SDD1306_128X64) {
169         #if defined(DISPLAY_TYPE_SDD1306_128X64)
170         u8x8_SSD1306->clear();
171         #endif
172     }
173 }
174
175 void Display::print(int vertical, int horizontal, const char text[]){
176     if (_screen == LCD_16X2 || _screen == LCD_16X2_I2C) {

```

```

177     #if defined(DISPLAY_TYPE_LCD_16X2) || (DISPLAY_TYPE_LCD_16X2_I2C)
178         lcd->setCursor(vertical, horizontal);
179         lcd->print(text);
180     #endif
181 }
182 else if (_screen == SDD1306_128X64) {
183     #if defined(DISPLAY_TYPE_SDD1306_128X64)
184         //u8x8_SSD1306->setCursor(vertical, (horizontal*2));
185         //u8x8_SSD1306->print(text);
186         u8x8_SSD1306->setCursor(vertical, (horizontal*2));
187         this->print(text);
188         //u8x8_SSD1306->display();
189     #endif
190 }
191 }
192
193 void Display::print(int line, const char text[]){
194     byte pos = _width -(strlen(text));
195     pos = (pos >> 1);
196     this->print((int)pos, line, text);
197 }
198
199 void Display::print(const char text[]){
200     if (_screen == LCD_16X2 || _screen == LCD_16X2_I2C) {
201         #if defined(DISPLAY_TYPE_LCD_16X2) || (DISPLAY_TYPE_LCD_16X2_I2C)
202             lcd->print(text);
203         #endif
204     }
205     else if (_screen == SDD1306_128X64) {
206         #if defined(DISPLAY_TYPE_SDD1306_128X64)
207             u8x8_SSD1306->print(text);
208             u8x8_SSD1306->flush();
209         #endif
210     }
211 }
212
213 void Display::print(const char text1[], const char text2[]){
214     if (_screen == LCD_16X2 || _screen == LCD_16X2_I2C) {
215         this->print(0, text1);
216         this->print(1, text2);
217     }
218     else if (_screen == SDD1306_128X64) {
219         this->print(1, text1);
220         this->print(2, text2);
221     }
222 }
223
224 void Display::print(const char text1[], const char text2[], const char text3[]){
225

```

```

226 }
227
228 void Display::print(const char text1[], const char text2[], const char text3[], const char text4[])
229
230 }
231
232 void Display::wait_anin(unsigned int t){
233     if (_screen == LCD_16X2 || _screen == LCD_16X2_I2C) {
234         #if defined(DISPLAY_TYPE_LCD_16X2) || (DISPLAY_TYPE_LCD_16X2_I2C)
235             lcd->setCursor(15,1);
236             lcd->write((byte)t%5);
237             #endif
238     }
239     else if (_screen == SDD1306_128X64) {
240         #if defined(DISPLAY_TYPE_SDD1306_128X64)
241             //char p = 0x2c;
242             u8x8_SSD1306->drawString((t%16),6,"-");
243             #endif
244     }
245 }
246 /*
247 void Display::draw_wait(byte t) {
248     #if defined(DISPLAY_TYPE_SDD1306_128X64)
249     byte draw_percet;
250     if (t == 0) draw_percet = 0b0000;
251     else if (t == 1) draw_percet = 0b0001;
252     else if (t == 2) draw_percet = 0b1001;
253     else if (t == 3) draw_percet = 0b1101;
254     else if (t == 4) draw_percet = 0b1111;
255     else if (t == 5) draw_percet = 0b1110;
256     else if (t == 6) draw_percet = 0b0110;
257     else if (t == 7) draw_percet = 0b0010;
258
259     u8x8_SSD1306->drawDisc(_width>>1,_height-8,7,draw_percet);
260     u8x8_SSD1306->drawCircle(_width>>1,_height-8,7,U8G2_DRAW_ALL);
261     #endif
262 }
263 */
264 void Display::print_PChar(byte c) {
265     if (_screen == LCD_16X2 || _screen == LCD_16X2_I2C) {
266         #if defined(DISPLAY_TYPE_LCD_16X2) || (DISPLAY_TYPE_LCD_16X2_I2C)
267             lcd->write(c);
268             #endif
269     }
270 }
271
272 void Display::splash(int time_delay){
273     // #if defined(DISPLAY_TYPE_SDD1306_128X64)
274     // u8x8_SSD1306->firstPage();

```

```
275     //do {
276     //endif
277     this->print(NAME, VERSION);
278     //if defined(DISPLAY_TYPE_SDD1306_128X64)
279     //} while( u8x8_SSD1306->nextPage() );
280     //endif
281     delay(time_delay);
282 }
```